In the United States Patent and Trademark Office
Board of Patent Appeals and Interferences

Appeal Brief

In re the Application of:

Margaret Ann Bernal, Christopher John Crone,
Andrei Fedorovich Lurie, and Daya Vivek

Serial No. 10/676,800
Filed: September 26, 2003
Attorney Docket No. SVL920030038US1

METHOD, SYSTEM, AND PROGRAM FOR OPTIMIZED PARAMETER BINDING

Submitted by:

Janaki K. Davda
Konrad, Raynes & Victor LLP
315 So. Beverly Dr., Ste. 210
Beverly Hills CA 90212
(310) 556-7983
(310) 556-7984 (fax)

TABLE OF CONTENTS

I. Real Party in Interest

The entire right, title and interest in this patent application is assigned to real party in interest International Business Machines Corporation.

II. Related Appeals, Interferences, and Judicial Proceedings

Appellant, Appellant's legal representative, and Assignee are not aware of any other prior or pending appeals, interferences, and judicial proceedings which may be related to, directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. Status of the Claims

Claims 1-3, 5-12, 14-21, 23-30, and 32-38 are pending and have been rejected in view of prior art. The final rejection of the claims is being appealed for all pending claims 1-3, 5-12, 14-21, 23-30, and 32-38.

IV. Status of Amendments

A response was filed on October 27, 2006 in response to a Final Office Action dated August 28, 2006, in which independent claims 1, 10, 19, and 28 were amended to incorporate language from corresponding dependent claims 4, 13, 22, and 31. Also, independent claims 37 and 38 were amended in a similar manner as claims 1, 10, 19, and 28.

In an Advisory Action (mailed on October 13, 2006), the Examiner indicated that claims 1-3, 5-12, 14-21, 23-30, and 32-38 stand rejected, but that amendments in the response filed on October 27, 2006 would be entered for purposes of appeal.

V. Summary of the Claimed Subject Matter

Claim 1 describes a method for input parameter binding (e.g., Specification, page 4, paragraph 11; page 22, paragraph 64; Figures 1, 10). When executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure (e.g., Specification, page 11, paragraph 35; Page 12, paragraph 39 – page 13, paragraph 40; Figs. 3 and 4). The application structure includes data to be inserted into a data store (e.g., Specification, page 8, paragraph 23). The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid (e.g., Specification, page 8, paragraph 20). When there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information (e.g., Specification, page 13, paragraphs 40-41).

Claim 10 describes a method for output parameter binding a method for input parameter binding (e.g., Specification, page 4, paragraph 12; page 22, paragraph 64; Figures 1, 10). When executing a statement, when performing bind-out of host variables, comparing data in an application structure received with the statement with optimization information in a bind-out structure (e.g., Specification, page 11, paragraph 35; Page 12, paragraph 39 – page 13, paragraph 40; Figs. 3 and 4). The application structure is capable of storing data to be retrieved from a data store (e.g., Specification, page 8, paragraph 23). The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid (e.g., Specification, page 8, paragraph 20). When there is a match between the data in the

application structure and data in the optimization information in the bind-out structure, executing the statement with the optimization information (e.g., Specification, page 13, paragraphs 40-41).

Claim 19 describes an article of manufacture including a program for input parameter binding, wherein the program causes operations to be performed (e.g., Specification, page 4, paragraph 11; page 22, paragraph 64; Figures 1, 10). When executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure (e.g., Specification, page 11, paragraph 35; Page 12, paragraph 39 – page 13, paragraph 40; Figs. 3 and 4). The application structure includes data to be inserted into a data store (e.g., Specification, page 8, paragraph 23). The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid (e.g., Specification, page 8, paragraph 20). When there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information (e.g., Specification, page 13, paragraphs 40-41).

Claim 28 describes an article of manufacture including a program for output parameter binding, wherein the program causes operations to be performed (e.g., Specification, page 4, paragraph 12; page 22, paragraph 64; Figures 1, 10). When executing a statement, when performing bind-out of host variables, comparing data in an application structure received with the statement with optimization information in a bind-out structure (e.g., Specification, page 11, paragraph 35; Page 12, paragraph 39 – page 13, paragraph 40; Figs. 3 and 4). The application structure is capable of storing data to be retrieved from a data store (e.g., Specification, page 8, paragraph 23). The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid (e.g., Specification, page 8, paragraph 20). When there is a match between the data in the application structure and data in the optimization information in the bind-out structure, executing the statement with the optimization information (e.g., Specification, page 13, paragraphs 40-41).

Claim 37 describes a system for input parameter binding (e.g., Specification, page 4, paragraph 11; page 22, paragraph 64; Figures 1, 10). When executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure (e.g., Specification, page 11, paragraph 35; Page 12, paragraph 39 – page 13, paragraph 40; Figs. 3 and 4). The application structure includes data to be inserted into a data store (e.g., Specification, page 8, paragraph 23). The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid (e.g., Specification, page 8, paragraph 20). When there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information (e.g., Specification, page 13, paragraphs 40-41).

Claim 38 describes a system for output parameter binding (e.g., Specification, page 4, paragraph 12; page 22, paragraph 64; Figures 1, 10). When executing a statement, when performing bind-out of host variables, comparing data in an application structure received with the statement with optimization information in a bind-out structure (e.g., Specification, page 11, paragraph 35; Page 12, paragraph 39 – page 13, paragraph 40; Figs. 3 and 4). The application structure is capable of storing data to be retrieved from a data store (e.g., Specification, page 8, paragraph 23). The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid (e.g., Specification, page 8, paragraph 20). When there is a match between the data in the application structure and data in the optimization information in the bind-out structure, executing the statement with the optimization information (e.g., Specification, page 13, paragraphs 40-41).

Claims 2, 11, 20, and 29 describe, when there is not a match between the data in the application structure and the optimization information, regenerating optimization information (e.g., Specification, page 13, paragraph 41).

Claims 3, 12, 21, and 30 describe, at bind time, storing the optimization information in the bind-in structure (e.g., Specification, page 10, paragraph 27).

Claims 5, 14, 23, and 32 describe, for fixed length data, storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a location of a next data value and calculating the location of the next data value by adding the increment length to the data pointer (e.g., Specification, page 19, paragraph 59).

Claims 6, 15, 24, and 33 describe, for distributed processing, at a client computer, calculating a location of data in a client communications buffer (e.g., Specification, page 15, paragraph 46; page 16, paragraph 49).

Claims 7, 16, 25, and 34 describe, for distributed processing, at a server computer, calculating a location of data in a server communications buffer (e.g., Specification, page 16, paragraph 50).

Claims 8, 17, 26, and 35 describe, for distributed processing, at a client computer, calculating a location of data in an application program address space (e.g., Specification, page 16, paragraph 48).

Claims 9, 18, 27, and 36 describe, when returning a handle to a cursor to a result set from a stored procedure to an application, recalculating the optimization information (e.g., Specification, page 20, paragraph 61).

VI. Grounds of Rejection to Be Reviewed on Appeal

A concise statement listing each ground of rejection presented for review is as follows:

A. Whether claims 1-3, 9-12, 18-21, 27-30, and 36-38 are unpatentable under 35 U.S.C. 103(a) over Kaluskar et al. (U.S. Patent No. 6,985,904) in view of Crone et al. (U.S. Patent No. 6,249,783).

B. Whether claims 5, 14, 23, and 32 are unpatentable under 35 U.S.C. 103(a) over Kaluskar et al. (U.S. Patent No. 6,985,904) in view of Crone et al. (U.S. Patent No. 6,249,783) and further in view of Desai et al. (U.S. Patent No. 6,567,816).

C. Whether claims 6-8, 15-17, 24-26, and 33-35 are unpatentable under 35 U.S.C. 103(a) over Kaluskar et al. (U.S. Patent No. 6,985,904) in view of Crone et al. (U.S. Patent No. 6,249,783) and further in view of Jordan II et al. (U.S. Patent No. 5,875,442).

VII. Argument

A. Rejection of Claims 1-3, 9-12, 18-21, 27-30, and 36-38 as unpatentable under 35 U.S.C. 103(a) over Kaluskar et al. (U.S. Patent No. 6,985,904) in view of Crone et al. (U.S. Patent No. 6,249,783)

1. Claims 1-3, 9-12, 18-21, 27-30, and 36-38

Claim 1 describes, when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure. The application structure includes data to be inserted into a data store. The optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid. When there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information.

As described in Applicants' Specification, page 2, paragraph 7:

> For multiple row inserts and fetches, some application programs insert or fetch one row of a table at a time, which increases execution costs. Some of the execution costs of an SQL statement to insert or fetch rows are incurred due to the DBMS determining the type of data present in the database and in the application program (e.g., an application program may describe data using a different type

than the database), finding out whether the database needs to convert the data from one encoding scheme (e.g., ASCII, EBCDIC, or Unicode) into another encoding scheme or from one data type (e.g., decimal or integer) to another data type, checking whether data conversion is valid (e.g., whether there is compatibility between the encoding schemes), getting information as to where to move data (i.e., the application program may change the source or target areas for each insert and fetch statement), and moving the data from the database to the application program or vice-versa. Thus, row by row inserts and fetches are expensive using existing parameter binding techniques because the DBMS cannot ensure that the application programs have not changed the data portion of their API from SQL statement to SQL statement, and, therefore, must determine information, such as data type, length, and encoding, during processing of each SQL statement.

Also, as described on page 12, paragraph 37, of the Specification:

If the optimization information stored in the bind-in and/or bind-out structures 152, 154, may be reused, the data store engine 130 does not need to look at how the application program is providing data (for insert) or how the application program wants data returned (for fetch). For example, if an application program fetched data from a table that contained integers, and the application program requested that the data be fetched into an array (e.g., a host-variable-array) of integers, then the bind-out logic of the bind-in and bind-out optimizer 132 would determine that the optimization information that was stored in the bind-out structure 156 at bind time could be used. If the application program fetched the data from the table containing integers into an array of small-integers, then the bind-out logic of the bind-in and bind-out optimizer 132 would determine that the bind time information could not be used, and the bind-in and bind-out optimizer 132 would recalculate the optimization information (e.g., data type, length, CCSID, array size, whether conversions are required, whether conversions are valid, and/or branch tags).

Applicants claimed invention provides a solution so that, when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, the statement is executed with the optimization information stored in the bind-in structure.

In the Office Action mailed on August 28, 2006, the Examiner submits that:

> When the method of Kaluskar compares the SQL statement, which is presently being processed, with an existing cursor, which corresponds to a first SQL statement (Kaluskar, Column 3 Line 65 through Column 4 Line 6), said method is effectively comparing "data in an application structure received with the statement (i.e., host variables) with optimization information" because said, method is comparing those host variable (i.e., data in an application structure received with the statement) with optimization information (i.e., matching an existing cursor, which corresponds to a first SQL statement, and, if said existing cursor exactly matches to the current SQL statement, reusing the execution plan of the executing cursor). Therefore, the method of Kaluskar does teach limitation of claim 1 "comparing data in an application structure received with the statement with optimization in a bind-structure".

Applicants respectfully traverse. For example, with reference to previously amended claim 1, rather than comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid, the Kaluskar patent compares a SQL statement with another SQL statement (Col. 4, line 2; FIG. 2, block 210 "matching SQL text"). For example, the Kaluskar patent at Col. 3, line 67 – Col. 4, line 3, describes that the matching step 210 of FIG. 2 is accomplished if an existing cursor corresponds to a

first SQL statement that exactly matches the SQL statement presently being processed. Also, the Kaluskar patent at Col. 4, lines 10-13, describes that two statements are considered "matching" in this embodiment at step 210 if no more than the values of one or more literals in the SQL statements are different. Thus, the Kaluskar patent is comparing SQL statements. Applicants respectfully submit that comparing SQL statements is not "effectively comparing" data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid.

The Kaluskar patent at Col. 2, line 62 – Col. 3, line 2, describes that a type checking stage engages data type resolution between a client process and a server process, which verifies and corrects data type incompatibilities that can exist, for example, in a heterogeneous enterprise client/server network. Also, the Crone patent at Col. 5, lines 24-57 describes methods to convert other primitive data types to the module's primitive data types. However, neither the Kaluskar patent nor the Crone patent, either alone or together, teaches or suggests comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid.

The law is well settled that a reference will not support a rejection based upon obviousness where the proposed modification to the reference contravenes the principle of operation of the device of the reference:

> If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims prima facie obvious. In re Ratti , 270 F.2d 810, 123 USPQ 349 (CCPA 1959)

The Examiner appears to be impermissibly modifying the Kaluskar patent's matching of SQL statements ("matching text") to a comparison of data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid. Applicants respectfully submit that such a modification changes the principle of operation of the Kaluskar patent.

Also, the Kaluskar patent describes that the objective of SQL statement compilation is the development of an execution plan (Col. 3, lines 3-4). The process of searching for a SQL statement is called a soft parse, which is preferred over a hard parse or compilation (Col. 1, lines 26-32; Col. 3, lines 61-64). The comparison of SQL statements is done during a soft parse because they are trying to reduce the expense of compilation involved in processing SQL statements (Col. 3, lines 44-45). The comparison of the Kaluskar patent occurs before compilation because, if there is no match, a hard parse is needed. Such processing before compilation teaches away from the *claimed comparison made when executing a statement*, as claimed by Applicants.

Because the Kaluskar patent does not teach or suggest the claimed comparison, Applicants respectfully submit that the Kaluskar patent does not teach or suggest, when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information. Also, the Kaluskar patent describes reusing an execution plan of an existing cursor to avoid performing a hard parse compilation (Col. 4, lines 3-6). On the other hand, rather than re-using an execution plan, the claimed invention executes the statement with the optimization information in the bind-in structure.

Moreover, Applicants respectfully submit that there is no teaching or suggestion in the Kaluskar patent of performing a comparison when performing bind-in of host variables.

Also, the Examiner states that "Kaluskar does not explicitly disclose that type conversion information is included in the recycled/reused execution plan of an existing

cursor. Therefore, Kaluskar does not explicitly teach the limitation 'when performing bind-in of host variables'", and cites the Crone patent as teaching this. Applicants respectfully traverse. The Crone patent describes methods to convert other primitive data types to the module's primitive data types (Col. 5, lines 39-41). However, the Crone patent does not cure the defects of the Kaluskar patent as the Crone patent does not teach or suggest, when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid.

Thus, claim 1 is not taught or suggested by the Kaluskar patent or the Crone patent, either alone or in combination.

Claims 19 and 37 are not taught or suggested by the Kaluskar patent or the Crone patent, either alone or in combination, for at least the same reasons as were discussed with respect to claim 1.

Claims 10, 28, and 38 describe bind-out, rather than bind-in (as described in claim 1). For example, claims 10, 28, and 38 also describe comparing data in an application structure received with the statement with optimization information in a bind-out structure, wherein the application structure is capable of storing data to be retrieved from a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid. Therefore, claims 10, 28, and 38 are not taught or suggested by the Kaluskar patent or the Crone patent, either alone or in combination, for at least the same reasons as were discussed with respect to claim 1.

Dependent claims 2-3, 9, 11-12, 18, 20-21, 27, 29-30, and 36 incorporate the language of independent claims 1, 10, 19, and 28 and add additional novel elements. Therefore, dependent claims 2-3, 9, 11-12, 18, 20-21, 27, 29-30, and 36 are not taught or

suggested by the Kaluskar patent or the Crone patent, either alone or in combination, for at least the same reasons as were discussed with respect to claims 1, 10, 19, and 28.

Accordingly, it is respectfully submitted that the rejection of claims 1-3, 9-12, 18-21, 27-30, and 36-38 as obvious over the Kaluskar and Crone combination should be reversed.

B. Rejection of Claims 5, 14, 23, and 32 as unpatentable under 35 U.S.C. 103(a) over Kaluskar et al. (U.S. Patent No. 6,985,904) in view of Crone et al. (U.S. Patent No. 6,249,783) and further in view of Desai et al. (U.S. Patent No. 6,567,816)

1. Claims 5, 14, 23, and 32

Applicants respectfully submit that the Desai patent does not cure the defects of the Kaluskar and Crone patents. For example, the Desai patent does not teach or suggest, when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid.

Dependent claims 5, 14, 23, and 32 incorporate the language of independent claims 1, 10, 19, and 28 and add additional novel elements. Thus, claims 5, 14, 23, and 32 are not taught or suggested by the Kaluskar patent, the Crone patent, or the Desai patent, either alone or in combination, for at least the same reasons as were discussed with respect to claims 1, 10, 19, and 28.

Accordingly, it is respectfully submitted that the rejection of claims 5, 14, 23, and 32 as obvious over the Kaluskar, Crone, and Desai combination should be reversed.

C. Rejection of Claims 6-8, 15-17, 24-26, and 33-35 as unpatentable under 35 U.S.C. 103(a) over Kaluskar et al. (U.S. Patent No. 6,985,904) in view of Crone et al. (U.S. Patent No. 6,249,783) and further in view of Jordan II et al. (U.S. Patent No. 5,875,442)

1. Claims 6-8, 15-17, 24-26, and 33-35

For example, the Jordan II patent does not cure the defects of the Kaluskar and Crone patents. For example, the Jordan II patent does not teach or suggest, when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid.

Dependent claims 6-8, 15-17, 24-26, and 33-35 incorporate the language of independent claims 1, 10, 19, and 28 and add additional novel elements. Thus, claims 6-8, 15-17, 24-26, and 33-35 are not taught or suggested by the Kaluskar patent, the Crone patent, or the Jordan II patent, either alone or in combination, for at least the same reasons as were discussed with respect to claims 1, 10, 19, and 28.

Accordingly, it is respectfully submitted that the rejection of claims 6-8, 15-17, 24-26, and 33-35 as obvious over the Kaluskar, Crone, and Jordan II combination should be reversed.

D. Conclusion

Each of the rejections set forth in the Final Office Action is improper and should be reversed.

Respectfully submitted,

Janaki K. Davda
Reg. No. 40,684
Konrad Raynes & Victor LLP
315 South Beverly Drive, Ste. 210
Beverly Hills, California 90212
Tel: 310-553-7977
Fax: 310-556-7984

Dated: January 29, 2007

VIII. Claims Appendix

1. (Previously Presented) A method for input parameter binding, comprising:

when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid; and

when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information.

2. (Original) The method of claim 1, further comprising:

when there is not a match between the data in the application structure and the optimization information, regenerating optimization information.

3. (Original) The method of claim 1, further comprising:

at bind time, storing the optimization information in the bind-in structure.

4. (Cancelled)

5. (Original) The method of claim 1, further comprising:

for fixed length data,

storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a location of a next data value; and

calculating the location of the next data value by adding the increment length to the data pointer.

6. (Original) The method of claim 1, further comprising:

for distributed processing, at a client computer, calculating a location of data in a client communications buffer.

7. (Original) The method of claim 1, further comprising:

for distributed processing, at a server computer, calculating a location of data in a server communications buffer.

8. (Original) The method of claim 1, further comprising:

for distributed processing, at a client computer, calculating a location of data in an application program address space.

9. (Original) The method of claim 1, further comprising:

when returning a handle to a cursor to a result set from a stored procedure to an application, recalculating the optimization information.

10. (Previously Presented) A method for output parameter binding, comprising:

when executing a statement, when performing bind-out of host variables, comparing data in an application structure received with the statement with optimization information in a bind-out structure, wherein the application structure is capable of storing data to be retrieved from a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid; and

when there is a match between the data in the application structure and data in the optimization information in the bind-out structure, executing the statement with the optimization information.

11. (Original) The method of claim 10, further comprising:

when there is not a match between the data in the application structure and the optimization information, regenerating optimization information.

12. (Original) The method of claim 10, further comprising:

at bind time, storing the optimization information in the bind-out structure.

13. (Cancelled)

14. (Original) The method of claim 10, further comprising:

for fixed length data,

storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a location of a next data value; and

calculating the location of the next data value by adding the increment length to the data pointer.

15. (Original) The method of claim 10, further comprising:

for distributed processing, at a client computer, calculating a location of data in a client communications buffer.

16. (Original) The method of claim 10, further comprising:

for distributed processing, at a server computer, calculating a location of data in a server communications buffer.

17. (Original) The method of claim 10, further comprising:

for distributed processing, at a client computer, calculating a location of data in an application program address space.

18. (Original) The method of claim 10, further comprising:

when returning a handle to a cursor to a result set from a stored procedure to an application, recalculating the optimization information.

19. (Previously Presented) An article of manufacture including a program for input parameter binding, wherein the program causes operations to be performed, the operations comprising:

when executing a statement, when performing bind-in of host variables, comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid; and

when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, executing the statement with the optimization information.

20. (Original) The article of manufacture of claim 19, wherein the operations further comprise:

when there is not a match between the data in the application structure and the optimization information, regenerating optimization information.

21. (Original) The article of manufacture of claim 19, wherein the operations further comprise:

at bind time, storing the optimization information in the bind-in structure.

22. (Cancelled)

23. (Original) The article of manufacture of claim 19, wherein the operations further comprise:

for fixed length data,

storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a next data value; and

calculating the location of the next data value by adding the increment length to the data pointer.

24. (Original) The article of manufacture of claim 19, wherein the operations further comprise:

for distributed processing, at a client computer, calculating a location of data in a client communications buffer.

25. (Original) The article of manufacture of claim 19, wherein the operations further comprise:

for distributed processing, at a server computer, calculating a location of data in a server communications buffer.

26. (Original) The article of manufacture of claim 19, wherein the operations further comprise:

for distributed processing, at a client computer, calculating a location of data in an application program address space.

27. (Original) The article of manufacture of claim 19,wherein the operations further comprise:

when returning a handle to a cursor to a result set from a stored procedure to an application, recalculating the optimization information.

28. (Previously Presented) An article of manufacture including a program for output parameter binding, wherein the program causes operations to be performed, the operations comprising:

when executing a statement, when performing bind-out of host variables, comparing data in an application structure received with the statement with optimization information in a bind-out structure, wherein the application structure is capable of storing data to be retrieved from a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an

indication of whether conversions are required, and an indication of whether the required conversions are valid; and

when there is a match between the data in the application structure and data in the optimization information in the bind-out structure, executing the statement with the optimization information.

29. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

when there is not a match between the data in the application structure and the optimization information, regenerating optimization information.

30. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

at bind time, storing the optimization information in the bind-out structure.

31. (Cancelled)

32. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

for fixed length data,

storing an increment length by which a data pointer that is pointing to data in an application program area is to be incremented to find a next data value; and

calculating the location of the next data value by adding the increment length to the data pointer.

33. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

for distributed processing, at a client computer, calculating a location of data in a client communications buffer.

34. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

for distributed processing, at a server computer, calculating a location of data in a server communications buffer.

35. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

for distributed processing, at a client computer, calculating a location of data in an application program address space.

36. (Original) The article of manufacture of claim 28, wherein the operations further comprise:

when returning a handle to a cursor to a result set from a stored procedure to an application, recalculating the optimization information.

37. (Previously Presented) A system for input parameter binding, comprising:

when executing a statement, when performing bind-in of host variables, means for comparing data in an application structure received with the statement with optimization information in a bind-in structure, wherein the application structure includes data to be inserted into a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid; and

when there is a match between the data in the application structure and data in the optimization information in the bind-in structure, means for executing the statement with the optimization information.

38. (Previously Presented) A system for output parameter binding, comprising:

when executing a statement, when performing bind-out of host variables, means for comparing data in an application structure received with the statement with

optimization information in a bind-out structure, wherein the application structure is capable of storing data to be retrieved from a data store and wherein the optimization information includes at least one of data type, length, Coded Character Set Identifier, an array size, an indication of whether conversions are required, and an indication of whether the required conversions are valid; and

when there is a match between the data in the application structure and data in the optimization information in the bind-out structure, means for executing the statement with the optimization information.

IX. Evidence Appendix

None

X. Related Proceedings Appendix

None